J. Math. Anal. Appl. 388 (2012) 873–887

Contents lists available at SciVerse ScienceDirect

Journal of Mathematical Analysis and Applications

www.elsevier.com/locate/jmaa

Coupling methods for heat transfer and heat flow: Operator splitting and the parareal algorithm

Jürgen Geiser [a,*], Stefan Güttel [b]

[a] Humboldt University of Berlin, Unter den Linden 6, D-10099 Berlin, Germany
[b] University of Oxford, England, United Kingdom

ARTICLE INFO

Article history:
Received 25 May 2011
Available online 3 November 2011
Submitted by W.L. Wendland

Keywords:
Heat transfer
Heat flux
Operator splitting
Iterative operator splitting
Parareal

ABSTRACT

We propose an operator splitting method for coupling heat transfer and heat flow equations. This work is motivated by the need to couple independent industrial heat transfer solvers (e.g., the Aura-Fluid software package) and heat flow solvers (e.g., Openfoam). Such packages are often used to simulate the influence of solar heat in car bodies and are coupled by A-B splitting techniques. The main goal of this work is the acceleration of the coupled software system by iterative operator splitting methods and additional time-parallelism using the parareal algorithm. We present these new splitting techniques along with some novel convergence results and test the splitting-parareal combination on various numerical problems.

1. Introduction

We are interested in the fast solution of coupled heat transfer and heat flow equations with splitting and time parallelization techniques. We concentrate on iterative splitting methods, which are related to iterative solver methods and have two historical origins:

- Picard–Lindelöf iterations [13,14], where waveform relaxation methods have their origins [20].
- Iterative split-operator approaches, see e.g. [10], wherein a two step method is derived to solve nonlinear reactive transport equations.

A detailed description along with the theoretical background of iterative splitting schemes is given in the monograph [8].

The main advantage of iterative splitting schemes is the possibility to obtain higher accuracy of order $\mathcal{O}(\tau^i)$ with additional iterative steps (where i denotes the number of iterations and τ is the length of the time-step). This is different to standard splitting schemes with fixed order, e.g. A-B splitting of order $\mathcal{O}(\tau)$ or Strang splitting of order $\mathcal{O}(\tau^2)$ (see [18]), where an increased accuracy can only be achieved through smaller time-steps τ, with the cost of an increased number of function evaluations, see [17].

Another advantage of (iterative) splitting schemes compared to the simultaneous solution of the full equation system is the possible simplification of the splitted equations representing decomposed operators, for which specialized solvers can be employed. For some practical problems, splitting techniques are unavoidable due to the fact that decoupled software packages specialized on different physical problems need to be combined. In the industrial application we have in mind (heat in car bodies), we have two different equations being solved with separate codes: while the heat transfer and radiation

* Corresponding author.
E-mail addresses: geiser@mathematik.hu-berlin.de (J. Geiser), stefan.guettel@maths.ox.ac.uk (S. Güttel).

0022-247X/$ – see front matter
doi:10.1016/j.jmaa.2011.10.030

is solved with the Aura software package, the flow field of the temperature is computed with a flow-field solver, e.g. Openfoam or Vectis, see [19].

These software packages also implement space parallelization by domain decomposition. However, in order to employ even more processors (with a number above the critical number of processors for which space parallelization is saturated) we make use of the so-called *parareal* iteration, see [15].

This paper is organized as follows. The general model is described in Section 2. The splitting methods related to the model equations are given in Section 3. In Section 4 we briefly review the parareal time-parallelization iteration used in Section 5, wherein we report on various numerical experiments with test problems.

2. Mathematical model

We consider a model of coupled heat transfer and heat flow. The heat transfer problem arises, for example, when the temperature distribution in a car body needs to be modeled over time, see [21], and therefore we have to apply large time-steps to simulate such problems.

The heat flow is given as a simplification of the inviscid compressible Navier–Stokes equation, see [16]. We deal with a inviscid flow based on the assumption that viscous forces are small in comparison to inertial forces. Such situations can be identified as flows with a Reynolds number much greater than one. The assumption that viscous forces are negligible can be used to simplify the Navier–Stokes equations to the Euler equations.

1. Heat transfer equation (Heat equation):

$$\begin{aligned} &\partial_t T = \nabla \cdot (K \nabla T) - \nabla \cdot (\mathbf{v} T), \\ &T(\mathbf{x}, t_0) = T_0(\mathbf{x}), \end{aligned} \tag{2.1}$$

where the unknown temperature is denoted by T. This equation contains a diffusion part and a convection part, both of which are typically treated as independent operators in splitting methods.

2. Heat flow equation (Euler equation):

$$\begin{aligned} &\partial_t \mathbf{v} = -(\mathbf{v} \cdot \nabla)\mathbf{v} - \nabla p(T), \\ &\mathbf{v}(\mathbf{x}, t_0) = \mathbf{v}_0(\mathbf{x}), \end{aligned} \tag{2.2}$$

where the unknown flux is $\mathbf{v}$ and we assume $\nabla \cdot \mathbf{v} = 0$. This equation contains a nonlinear flow part and a pressure part, both of which are typically treated as different, independent operators in splitting methods.

However, in this paper we are not mainly concerned with decomposing the equations themselves, but rather in the coupling of both equations to each other by an iterative splitting scheme, see [12].

Remark 2.1. For simplicity, we sometimes (e.g., in the first of our numerical experiments) assume that p is independent of T, in which case we obtain a uni-directional coupling between (2.2) and (2.1). This means that one can solve (2.2) and then use the result directly for solving (2.1).

Remark 2.2. The coupling of both equations becomes more involved when the dependence between them is strong or highly nonlinear. In particular, even the evaluation of $\nabla p(T)$ can become difficult if the pressure is only given implicitly. Therefore the idea is to apply an iterative splitting scheme, which couples the two delicate equations via a relaxation scheme, see [7]. With this as an advantage, we can do better than a non-iterative scheme that has stability problems, while we deal with stiff problems, see [17]. Here we use the smoothing property of the iterative schemes and overcome the stiffness problem, see [8].

3. Splitting methods

In the following we briefly discuss the coupling methods used for the heat transfer equation.

3.1. Lie–Trotter or A-B splitting method

The simplest scheme (and probably the one implemented most often) is the so-called A-B splitting method:

$$\partial_t \mathbf{v} = -(\mathbf{v} \cdot \nabla)\mathbf{v}, \quad \text{with } t^n \leqslant t \leqslant t^{n+1},\ \mathbf{v}\big(\mathbf{x}, t^n\big) = \mathbf{v}^n(\mathbf{x}), \tag{3.1}$$

$$\frac{\partial T}{\partial t} = \nabla \cdot (K \nabla T) - \nabla \cdot \tilde{\mathbf{v}} T, \quad \text{with } t^n \leqslant t \leqslant t^{n+1},\ \tilde{\mathbf{v}}\big(\mathbf{x}, t^n\big) = \mathbf{v}^{n+1}(\mathbf{x}),\ T\big(\mathbf{x}, t^n\big) = T\big(\mathbf{x}, t^n\big) \tag{3.2}$$

where T^n is the known initial value of the previous solution and $T(t^{n+1}) = T_2(\mathbf{x}, t^{n+1})$ is the approximate solution of the full equation.

This method results in a global splitting error $O(\Delta t)$, where Δt denotes the length of the time-step.

3.2. Strang splitting

The Strang splitting method is given by the following equations:

$$\frac{\partial T_1(x,t)}{\partial t} = \nabla\cdot(K\nabla T_1) - \nabla\cdot\mathbf{v}_1 T_1, \quad \text{with } t^n \leqslant t \leqslant t^{n+1/2},\ \mathbf{v}_1\big(\mathbf{x},t^n\big) = \mathbf{v}^n(\mathbf{x}),\ T_1\big(\mathbf{x},t^n\big) = T\big(\mathbf{x},t^n\big), \tag{3.3}$$

$$\partial_t \mathbf{v}_2 = -(\mathbf{v}_2\cdot\nabla)\mathbf{v}_2, \quad \text{with } t^n \leqslant t \leqslant t^{n+1},\ \mathbf{v}_2\big(\mathbf{x},t^n\big) = \mathbf{v}^n(\mathbf{x}), \tag{3.4}$$

$$\frac{\partial T_3(x,t)}{\partial t} = \nabla\cdot(K\nabla T_3) - \nabla\cdot\mathbf{v}_3 T_3,$$
$$\text{with } t^{n+1/2} \leqslant t \leqslant t^{n+1},\ \mathbf{v}_3\big(\mathbf{x},t^{n+1/2}\big) = \mathbf{v}_2^{n+1}(\mathbf{x}),\ T_3(x,t_{n+1/2}) = T_1\big(x,t^{n+1/2}\big), \tag{3.5}$$

where T^n is the known initial value of the previous solution and $T(t^{n+1}) = T_3(x,t^{n+1})$ is the approximate solution of the full equation.

This method has a global splitting error of order $O(\Delta t^2)$.

3.3. Iterative splitting method

3.3.1. Linear case

The following algorithm is based on an iteration with fixed splitting discretization step-size τ [1]. On the time interval $[t^n, t^{n+1}]$ we solve the following subproblems consecutively for $i = 0, 2, \ldots, 2m$:

$$\partial_t \mathbf{v}_i = -(\mathbf{v}_i\cdot\nabla)\mathbf{v}_i, \quad \text{with } t^n \leqslant t \leqslant t^{n+1},\ \mathbf{v}_i\big(\mathbf{x},t^n\big) = \mathbf{v}^n(\mathbf{x}), \tag{3.6}$$

$$\frac{\partial T_i}{\partial t} = \nabla\cdot(K\nabla T_i) - \nabla\cdot\mathbf{v}_{i-1} T_i, \quad \text{with } t^n \leqslant t \leqslant t^{n+1},\ \mathbf{v}_{i-1}\big(\mathbf{x},t^n\big) = \mathbf{v}^n(\mathbf{x}),\ T\big(\mathbf{x},t^n\big) = T\big(\mathbf{x},t^n\big) \tag{3.7}$$

where $T^n, \mathbf{v}^n$ is the split approximation at time $t = t^n$ [1].

Here we solve the time-discretization with a backward differential formula of fourth order (BDF4 method), see [9].

The higher order is obtained by applying recursively a fixed-point iteration to reconstruct the analytical solution of the coupled operators, see [5].

3.3.2. Generalization to systems of ODEs

We deal with a vectorial iterative scheme, given as an inner and outer iterative scheme.

While the outer iterative scheme is a Waveform relaxation scheme and could be seen as a coarse iterative scheme, since we iterate over the full system in one step. The inner iterative scheme is a multi-iterative Waveform relaxation scheme: it iterates over each ODE in m steps.

Outer iteration (Waveform relaxation, iteration over the full system):

$$\frac{dU_i}{dt} = \mathcal{A}U_i + \mathcal{B}U_{i-1} + F, \tag{3.8}$$

$$U_i\big(t^n\big) = U\big(t^n\big), \tag{3.9}$$

$$i = 1, \ldots, I, \tag{3.10}$$

where $U_i = (u_{1,i}, \ldots, u_{m,i})$ and m is the number of ODEs. Furthermore,

$$\mathcal{A} = \begin{pmatrix} A_{1,1} & A_{1,2} & \ldots & A_{1,m} \\ A_{2,1} & A_{2,2} & \ldots & A_{2,m} \\ \vdots & & & \\ A_{m,1} & A_{m,2} & \ldots & A_{m,m} \end{pmatrix}, \tag{3.11}$$

and

$$\mathcal{B} = \begin{pmatrix} B_{1,1} & B_{1,2} & \ldots & B_{1,m} \\ B_{2,1} & B_{2,2} & \ldots & B_{2,m} \\ \vdots & & & \\ B_{m,1} & B_{m,2} & \ldots & B_{m,m} \end{pmatrix}, \tag{3.12}$$

are matrices of the system of ODEs, for example the diagonal and outer-diagonal matrices and F is a right hand side (e.g., a source term).

Inner iteration (iterative splitting, relaxation over each sub-equation):

$$\frac{dU_{1,j_1}}{dt} = A_{1,1}U_{1,j_1} + A_{1,2}U_{1,j_1-1} + \cdots + A_{1,m}U_{m,j_1-1} + B_{1,1}U_{1,j_1-1} + B_{1,2}U_{1,j_1-1} + \cdots + B_{1,m}U_{m,j_1-1} + f,$$
$$U_{1,j_1}(t^n) = U_1(t^n), \quad j_1 = 1,\ldots,J_1, \tag{3.13}$$
$$\frac{dU_{2,j_2}}{dt} = A_{2,1}U_{1,j_2-1} + A_{2,2}U_{1,j_2} + \cdots + A_{2,m}U_{m,j_2-1} + B_{2,1}U_{1,j_2-1} + B_{2,2}U_{1,j_2-1} + \cdots + B_{2,m}U_{m,j_2-1} + f,$$
$$U_{2,j_2}(t^n) = U_2(t^n), \quad j_2 = 1,\ldots,J_2, \tag{3.14}$$
$$\ldots$$
$$\frac{dU_{m,j_m}}{dt} = A_{m,1}U_{1,j_m-1} + A_{m,2}U_{1,j_m-1} + \cdots + A_{m,m}U_{m,j_m} + B_{m,1}U_{1,j_m-1} + B_{m,2}U_{1,j_m-1} + \cdots + B_{m,m}U_{m,j_m-1} + f,$$
$$U_{m,j_m}(t^n) = U_m(t^n), \quad j_m = 1,\ldots,J_m, \tag{3.15}$$

where $U_i(t^{n+1}) = (U_{1,J_1}(t^{n+1}),\ldots,U_{m,J_m}(t^{n+1}))$ is the result for the next iterated step i and $U_j(t^n) = (U_1(t^n),\ldots,U_m(t^n))$ is the initial solution. $(U_{1,j_1-1}(t^{n+1}),\ldots,U_{m,j_m-1}(t)) = U_{i-1}(t)$ is the approximate starting solution to $i-1$ of the outer iterative step.

Remark 3.1. Based on the iterative scheme of the inner iteration (3.13)–(3.15), we iterative over the suboperators $A_{1,1},\ldots,A_{m,m}$, while the suboperators $B_{1,1},\ldots,B_{m,m}$ can be applied explicitly, meaning as a right hand side. If we also include the suboperators $B_{1,1},\ldots,B_{m,m}$ as implicit operators in the scheme, meaning that we also iterate over such operators, we obtain finer schemes with more accurate results, see [7].

3.3.3. Unifying analysis

Application of an alternative waveform relaxation scheme:

1) Convergence of the inner iterations.

2) Convergence of the outer iterations.

The analysis is based on the convergence result of each inner iteration scheme that is equal to the order of the outer iteration scheme, or one order higher.

If all inner schemes converge and are at least of the same order as the outer schemes, then the full iterative scheme is convergent.

Theorem 3.1. *Let us consider the abstract Cauchy problem in a Banach space* $\mathbf{X}$ *in finite dimensions with Banach norm* $\|\cdot\|_{\mathbf{X}} = \|\cdot\|$. *The time-step is* $\tau = t^{n+1} - t^n$, *with* $t \in [t^n, t^{n+1}]$.

$$\frac{dU}{dt} = \mathcal{A}U + \mathcal{B}U + F, \quad t \in \left(t^n, t^{n+1}\right), \tag{3.16}$$
$$U(t^n) = U_n, \tag{3.17}$$

where $U(t^{n+1}) = (u_1(t^{n+1}),\ldots,u_m(t^{n+1}))$ *is the solution at time* t^{n+1}, *and* m *is the number of ODEs. Furthermore, the matrices are given by* (3.11) *and* (3.12) *and* F *is a right hand side (e.g., a source term).*

We wish to obtain an accuracy of $\mathcal{O}(\tau)$ *with* $\tilde{j} = \min\{J_1,\ldots,J_m\}$ *inner iterative steps, while* $J_1,\ldots,J_m$ *are the subiterative steps in each subiterative process.*

Then the iteration process (3.8) *is convergent with order* $\mathcal{O}(\tau^i)$, *where* i *is the number of steps in the outer iteration.*

Proof. The outer iterative process satisfies the convergence bound

$$\left\|E_i(t)\right\| \leqslant K\tau_n \left\|E_{i-1}(t)\right\|, \tag{3.18}$$

where E_i is the ith error $E_i(t) := U(t) - U_i(t)$ and $U(t)$ denotes the exact solution of the ODE. The proof of this assertion is given in [5].

Further the inner iterative process satisfies

$$\left\|E_i(t)\right\| \leqslant K\tau_n \left\|E_{\tilde{j}}(t)\right\|, \tag{3.19}$$

where E_i is ith error $E_i(t) := U(t) - U_i(t)$ and $U(t)$ is the given exact solution of the ODE.

Further we have assumed that $\|E_{\tilde{j}}(t)\| := \|U(t) - U_{\tilde{j}}(t)\| = \mathcal{O}(\tau^i)$ and $\tilde{j} = \min\{J_1,\ldots,J_m\}$ is the minimum number of iterative steps needed to obtain an accuracy of $\mathcal{O}(\tau^i)$. This means that the results of the inner iterative scheme are as accurate as the results of the outer scheme.

So we need at least a minimum of one iterative step over each single equation to gain at least one order of accuracy for the full system. □

3.4. Application to the heat transfer and heat flow equations

We have the following equation schemes:

$$\partial_t T_i = \nabla \cdot (K \nabla T_i) - \nabla \cdot \mathbf{v}_{i-1} T_{i-1}, \tag{3.20}$$

$$\partial_t \mathbf{v}_i = -(\mathbf{v}_{i-1} \cdot \nabla)\mathbf{v}_i - \nabla p(T_{i-1}), \tag{3.21}$$

$$T_i\big(\mathbf{x}, t^n\big) = T\big(\mathbf{x}, \mathbf{t^n}\big),$$

$$\mathbf{v}_i\big(\mathbf{x}, t^n\big) = \mathbf{v}\big(\mathbf{x}, t^n\big), \quad \text{for } t \in \big[t^n, t^{n+1}\big],\ n = 0, 1, 2, \ldots, N, \text{ with } i = 1, 2, \ldots, I,$$

where the initialization (starting condition for $i = 0$) is $T_0(\mathbf{x}, t) = T(\mathbf{x}, \mathbf{t^n})$ and $\mathbf{v}_0(\mathbf{x}, t) = \mathbf{v}(\mathbf{x}, \mathbf{t^n})$ with $t \in [t^n, t^{n+1}]$, which means the solution at the last time-point.

3.4.1. Nonlinear case: Modified Jacobian–Newton methods and fixpoint-iteration methods

We describe the modified Jacobian–Newton methods and fixpoint-iteration methods.

For weak nonlinearities, e.g., a quadratic nonlinearity, we propose the fixpoint iteration method, where our iterative operator splitting method is one, see [10]. For stronger nonlinearities, e.g., cubic or higher-order polynomial nonlinearities, the modified Jacobian method with embedded iterative splitting methods is suggested.

The point of embedding the splitting methods into the Newton methods is to decouple the equation system into simpler equations. Such simpler systems of equations can be solved with scalar Newton methods.

3.4.1.1. The altered Jacobian–Newton iterative methods with embedded sequential splitting methods. We confine our attention to time-dependent partial differential equations of the form

$$\frac{du}{dt} = A\big(u(t)\big)u(t) + B\big(u(t)\big)u(t), \quad \text{with } u\big(t^n\big) = u^n, \tag{3.22}$$

where $A(u), B(u) : \mathbf{X} \to \mathbf{X}$ are linear and densely defined in the real Banach space $\mathbf{X}$, involving only spatial derivatives of c, see [22]. We assume also that we have a weakly nonlinear operator with $A(u)u = \lambda_1 u$ and $B(u)u = \lambda_2 u$, where λ_1 and λ_2 are constant factors.

In the following we discuss the embedding of a sequential splitting method into the Newton method.

The altered Jacobian–Newton iterative method with an embedded iterative splitting method is given by Newton's method:

$F(u) = \frac{du}{dt} - A(u(t))u(t) - B(u(t))u(t)$ and we can compute $u^{(k+1)} = u^{(k)} - D(F(u^{(k)}))^{-1} F(u^{(k)})$, where $D(F(u))$ is the Jacobian matrix and $k = 0, 1, \ldots$.

We stop the iterations when we obtain $|u^{(k+1)} - u^{(k)}| \leqslant err$, where err is an error bound, e.g., $err = 10^{-4}$.

We assume a spatial discretization, with spatial grid points, $i = 1, \ldots, m$, and obtain the system of differential equations

$$F(u) = \begin{pmatrix} F(u_1) \\ F(u_2) \\ \vdots \\ F(u_m) \end{pmatrix}, \tag{3.23}$$

where $u = (u_1, \ldots, u_m)T$ and m is the number of spatial grid points.

The Jacobian matrix for the equation system is

$$DF(u) = \begin{pmatrix} \frac{\partial F(u_1)}{u_1} & \frac{\partial F(u_1)}{u_2} & \cdots & \frac{\partial F(u_1)}{u_m} \\ \frac{\partial F(u_2)}{u_1} & \frac{\partial F(u_2)}{u_2} & \cdots & \frac{\partial F(u_2)}{u_m} \\ \vdots & & & \\ \frac{\partial F(u_m)}{u_1} & \frac{\partial F(u_m)}{u_2} & \cdots & \frac{\partial F(u_m)}{u_m} \end{pmatrix},$$

where $u = (u_1, \ldots, u_m)$.

The modified Jacobian is

$$DF(u) = \begin{pmatrix} \frac{\partial F(u_1)}{u_1} + F(u_1) & \frac{\partial F(u_1)}{u_2} & \cdots & \frac{\partial F(u_1)}{u_m} \\ \frac{\partial F(u_2)}{u_1} & \frac{\partial F(u_2)}{u_2} + F(u_2) & \cdots & \frac{\partial F(u_2)}{u_m} \\ \vdots & & & \\ \frac{\partial F(u_m)}{u_1} & \frac{\partial F(u_m)}{u_2} & \cdots & \frac{\partial F(u_m)}{u_m} + F(u_m) \end{pmatrix},$$

where $u = (u_1, \ldots, u_n)$.

By embedding the sequential splitting method, we obtain the following algorithm:
We decouple the original system into two systems of equations:

$$F_1(u_1) = \partial_t u_1 - A(u_1)u_1 = 0 \quad \text{with } u_1(t^n) = u^n, \tag{3.24}$$

$$F_2(u_2) = \partial_t u_2 - B(u_2)u_2 = 0 \quad \text{with } u_2(t^n) = u_1(t^{n+1}), \tag{3.25}$$

where the results of the methods are $u(t^{n+1}) = u_2(t^{n+1})$ and $u_1 = (u_{11}, \ldots, u_{1n})$, $u_2 = (u_{21}, \ldots, u_{2n})$.

Thus we have to implement two Newton methods, each on one system of equations. The contribution is to reduce the Jacobian matrix to a diagonal matrix, e.g., with a weighted Newton method, see [12]. The splitting method with embedded Newton method is:

Algorithm 3.1. We assume the spatial operators A and B are discretized, e.g., we use finite difference or finite element methods. Furthermore, all initial conditions and boundary conditions are given discretely. Then we can apply Newton's method in its discrete form:

$$u_1^{(k+1)} = u_1^{(k)} - D\big(F_1\big(u_1^{(k)}\big)\big)^{-1}\big(\partial_t u_1^{(k)} - A\big(u_1^{(k)}\big)u_1^{(k)}\big), \tag{3.26}$$

$$\text{with } D\big(F_1\big(u_1^{(k)}\big)\big) = \frac{\partial}{\partial u_1^{(k)}}\big(\partial_t u_1^{(k)} - A\big(u_1^{(k)}\big)u_1^{(k)}\big), \tag{3.27}$$

$$u_1^{(k)}(t^n) = u^n \text{ and } k = 0, 1, 2, \ldots, K, \tag{3.28}$$

$$u_2^{(l+1)} = u_2^{(l)} - D\big(F_2\big(u_2^{(l)}\big)\big)^{-1}\big(\partial_t u_2^{(l)} - B\big(u_2^{(l)}\big)u_2^{(l)}\big), \tag{3.29}$$

$$\text{with } D\big(F_2\big(u_2^{(l)}\big)\big) = \frac{\partial}{\partial u_1^{(l)}}\big(\partial_t u_2^{(l)} - B\big(u_2^{(l)}\big)u_2^{(l)}\big), \tag{3.30}$$

$$u_2^{(l)}(t^n) = u_1^K(t^{n+1}) \text{ and } l = 0, 1, 2, \ldots, L, \tag{3.31}$$

where k and l are the iteration indices, and K and L are the maximal iterative steps for each part of the Newton method. These maximal iterative steps allow us to have at most an error of $|u_1^{(K)(t^{n+1})} - u_1^{(K-1)(t^{n+1})}| \leqslant err$, and

$$\big|u_2^{(L)(t^{n+1})} - u_2^{(L-1)(t^{n+1})}\big| \leqslant err,$$

where err is the error bound, e.g., $err = 10^{-6}$.

The approximate solution is then

$$u(t^{n+1}) = u_2^{(L)(t^{n+1})}.$$

For the improved method, we can apply the weighted Newton method. We try to skip the delicate outer diagonals in the Jacobian matrix and apply:

$$u_1^{(k+1)} = u_1^{(k)} - \big(D\big(F_1\big(u_1^{(k)}\big)\big) + \delta_1\big(u_1^{(k)}\big)\big)^{-1}\big(F_1\big(u_1^{(k)}\big) + \epsilon u_1^{(k)}\big), \tag{3.32}$$

where the function δ can be applied as a scalar, e.g., $\delta = 10^{-6}$, and the same with ϵ. It is important to ensure that δ is small enough to preserve the convergence.

Remark 3.2. If we assume that we discretize Eqs. (3.24) and (3.25) with the backward-Euler method, e.g.,

$$F_1\big(u_1(t^{n+1})\big) = u_1(t^{n+1}) - u_1(t^n) - \Delta t A\big(u_1(t^{n+1})\big)u_1(t^{n+1}) = 0 \quad \text{with } u_1(t^n) = c^n,$$

$$F_2(u_2) = u_2(t^{n+1}) - u_2(t^n) - \Delta t B\big(u_2(t^{n+1})\big)u_2(t^{n+1}) = 0 \quad \text{with } u_2(t^n) = u_1(t^{n+1}),$$

then we obtain the derivatives $D(F_1(u_1(t^{n+1})))$ and $D(F_2(u_2(t^{n+1})))$:

$$D\big(F_1\big(u_1(t^{n+1})\big)\big) = 1 - \Delta t\left(A\big(u_1(t^{n+1})\big) + \frac{\partial A(u_1(t^{n+1}))}{\partial u_1(t^{n+1})}u_1(t^{n+1})\right),$$

$$D\big(F_2(u_2)\big) = 1 - \Delta t\left(B\big(u_2(t^{n+1})\big) + \frac{\partial B(u_2(t^{n+1}))}{\partial u_2(t^{n+1})}u_2(t^{n+1})\right).$$

We can apply Eq. (3.32) analogously to $u_2^{(l+1)}$.

3.4.1.2. Iterative operator-splitting method as a fixpoint scheme. The iterative operator-splitting method can be used as a fixpoint scheme to linearize nonlinear operators, see [5] and [10].

We confine our attention to time-dependent partial differential equations of the form

$$\frac{du}{dt} = A\big(u(t)\big)u(t) + B\big(u(t)\big)u(t), \quad \text{with } u\big(t^n\big) = u^n, \tag{3.33}$$

where $A(u), B(u) : \mathbf{X} \to \mathbf{X}$ are nonlinear and densely defined in the real Banach space $\mathbf{X}$, and involve only spatial derivatives of c, see [22]. In the following we discuss the standard iterative operator-splitting methods as fixpoint iteration methods to linearize these operators.

We split our nonlinear differential equation (3.33) by applying

$$\frac{du_i(t)}{dt} = A\big(u_{i-1}(t)\big)u_i(t) + B\big(u_{i-1}(t)\big)u_{i-1}(t), \quad \text{with } u_i\big(t^n\big) = u^n, \tag{3.34}$$

$$\frac{du_{i+1}(t)}{dt} = A\big(u_{i-1}(t)\big)u_i(t) + B\big(u_{i-1}(t)\big)u_{i+1}, \quad \text{with } u_{i+1}\big(t^n\big) = u^n, \tag{3.35}$$

where the time-step is $\tau = t^{n+1} - t^n$. The iterations are $i = 1, 3, \ldots, 2m+1$. $u_0(t) = u_n$ is the starting solution, where we assume the solution u^{n+1} is near c^n, or $u_0(t) = 0$. So we have to solve the local fixpoint problem. u^n is the known split approximation at the time level $t = t^n$.

The split approximation at time level $t = t^{n+1}$ is defined by $u^{n+1} = u_{2m+2}(t^{n+1})$. We assume the operators $A(u_{i-1})$, $B(u_{i-1}) : \mathbf{X} \to \mathbf{X}$ are linear and densely defined on the real Banach space $\mathbf{X}$ for $i = 1, 3, \ldots, 2m+1$.

Here the linearization is done by using the previous iterative solution, such that $A(u_{i-1}), B(u_{i-1})$ are at least non-dependent operators in the iterative equations, and we can apply the linear theory.

The linearization is at least in the first equation $A(u_{i-1}) \approx A(u_i)$, and in the second equation $B(u_{i-1}) \approx B(u_{i+1})$.

We have

$$\big\| A\big(u_{i-1}\big(t^{n+1}\big)\big)u_i(t^{n+1}) - A\big(u^{n+1}\big)u\big(t^{n+1}\big)\big\| \leqslant \epsilon,$$

with sufficient iterations $i = \{1, 3, \ldots, 2m+1\}$.

Remark 3.3. The linearization with the fixpoint scheme can be used for smooth or weakly nonlinear operators, but otherwise we lose the convergence behavior, while we did not converge to the local fixpoint, see [10].

The second idea is based on the Newton method.

3.4.1.3. Jacobian–Newton iterative method with embedded operator-splitting method. The Newton method is used to solve the nonlinear parts of the iterative operator-splitting method (see the linearization techniques in [10,11]).

Newton method:

The function is

$$F(u) = \frac{\partial u}{\partial t} - A\big(u(t)\big)u(t) - B\big(u(t)\big)u(t) = 0.$$

The iteration can be computed by:

$$u^{(k+1)} = u^{(k)} - D\big(F\big(u^{(k)}\big)\big)^{-1} F\big(u^{(k)}\big),$$

where $D(F(u))$ is the Jacobian matrix and $k = 0, 1, \ldots$. Here $u = (u_1, \ldots, u_m)$ is the solution vector of the spatially discretized nonlinear equation.

We next have to apply the iterative operator-splitting method and then obtain

$$F_1(u_i) = \partial_t u_i - A(u_i)u_i - B(u_{i-1})u_{i-1} = 0, \tag{3.36}$$

$$\text{with } u_i\big(t^n\big) = u^n, \tag{3.37}$$

$$F_2(u_{i+2}) = \partial_t u_{i+1} - A(u_i)u_i - B(u_{i+1})u_{i+1} = 0, \tag{3.38}$$

$$\text{with } u_{i+1}\big(t^n\big) = u^n, \tag{3.39}$$

where the time-step is $\tau = t^{n+1} - t^n$. The iterations are $i = 1, 3, \ldots, 2m+1$. $u_0(t) = 0$ is the starting solution and u^n is the known split approximation at the time-level $t = t^n$. The results of these methods are $u(t^{n+1}) = u_{2m+2}(t^{n+1})$.

Thus we have to apply the Newton method twice and the contribution will be to reduce the Jacobian matrix, e.g., skip the diagonal entries. The splitting method with the embedded Newton method is

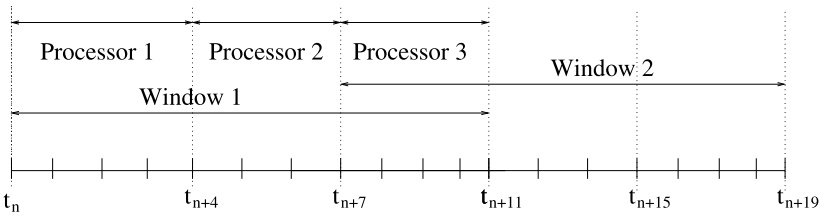

Fig. 4.1. Parallelization with parareal, windowing of the parallel process.

$$u_i^{(k+1)} = u_i^{(k)} - D\big(F_1\big(u_i^{(k)}\big)\big)^{-1}\big(\partial_t u_i^{(k)} - A\big(u_i^{(k)}\big)u_i^{(k)} - B\big(u_{i-1}^{(k)}\big)u_{i-1}^{(k)}\big),$$

$$\text{with } D\big(F_1\big(u_i^{(k)}\big)\big) = -\left(A\big(u_i^{(k)}\big) + \frac{\partial A(u_i^{(k)})}{\partial u_i^{(k)}}u_i^{(k)}\right), \text{ and } k = 0, 1, 2, \ldots, K, \text{ with } u_i\big(t^n\big) = u^n,$$

$$u_{i+1}^{(l+1)} = u_{i+1}^{(l)} - D\big(F_2\big(u_{i+1}^{(l)}\big)\big)^{-1}\big(\partial_t u_{i+1}^{(l)} - A\big(u_i^{(k)}\big)u_i^{(k)} - B\big(u_{i+1}^{(k)}\big)u_{i+1}^{(k)}\big),$$

$$\text{with } D\big(F_2\big(u_{i+1}^{(l)}\big)\big) = -\left(B\big(u_{i+1}^{(l)}\big) + \frac{\partial B(u_{i+1}^{(l)})}{\partial u_{i+1}^{(l)}}u_{i+1}^{(l)}\right), \text{ and } l = 0, 1, 2, \ldots, L, \text{ with } u_{i+1}\big(t^n\big) = u^n,$$

where the time-step is $\tau = t^{n+1} - t^n$. The iterations are $i = 1, 3, \ldots, 2m+1$. $u_0(t) = 0$ is the starting solution and c^n is the known split approximation at the time-level $t = t^n$. The results of the methods are $u(t^{n+1}) = u_{2m+2}(t^{n+1})$.

To get the improvement by skipping the delicate outer diagonals in the Jacobian matrix, we apply $u_i^{(k+1)} = u_i^{(k)} - (D(F_1(u_i^{(k)})) + \delta_1(u_i^{(k)}))^{-1}(F_1(u_i^{(k)}) + \epsilon u_i^{(k)})$, and analogously $u_{i+1}^{(l+1)}$.

Remark 3.4. For the iterative operator-splitting method with Newton iteration we have two iteration procedures. The first iteration is the Newton method to compute the solution of the nonlinear equations, and the second iteration is the iterative splitting method, which computes the resulting solution of the coupled equation systems. This embedded method is used for strong nonlinearities.

4. Parallelization: parareal

To improve the parallel scaling properties of our algorithm, we employ the so-called parareal algorithm invented by Lions, Maday and Turincini [15]. This algorithm is a means of time-parallelization and can be viewed as a multiple shooting method [4,3,2].

We assume we have a partition of the time interval, i.e., $\Omega = [0, T]$ divided into N subdomains:

$$\Omega_n = [T_{n-1}, T_n], \quad n = 1, 2, \ldots, N. \tag{4.1}$$

In parareal, one makes use of two solvers: a coarse propagator $G(T_n, T_{n-1}, x)$ and a fine propagator $F(T_n, T_{n-1}, x)$, which compute coarse and fine approximations of the solution U_n of the equation

$$\frac{du}{dt} = f\big(t, u(t)\big), \quad \text{with } u(T_{n-1}) = x. \tag{4.2}$$

It is crucial that the coarse integrator is computationally much faster than the fine integrator, the latter of which is more accurate. The first iteration of parareal uses the coarse integrator in a serial fashion to provide initial conditions to each time-slice Ω_n:

$$u_n^1 = G\big(T_n, T_{n-1}, u_{n-1}^1\big), \quad n = 1, 2, \ldots, N.$$

After this initialization the fine propagator can be used to integrate independently (i.e., in parallel) N initial-value problems $F(T_n, T_{n-1}, u_{n-1}^k)$ $(n = 1, 2, \ldots, N)$, yielding new approximations for the initial conditions on the following time-slices. In each iteration k, the corrections are then again quickly propagated using the coarse integrator:

$$u_n^{k+1} = F\big(T_n, T_{n-1}, u_{n-1}^k\big) + G\big(T_n, T_{n-1}, u_{n-1}^{k+1}\big) - G\big(T_n, T_{n-1}, u_{n-1}^k\big). \tag{4.3}$$

Example 4.1. We assume we have F as the iterative splitting propagator and G as the A-B splitting propagator.

Furthermore, the iterative splitting scheme also includes a fixpoint scheme for nonlinear problems.

So we step by each window to the next time interval, see Fig. 4.1.

5. Numerical experiments

In the following we treat some benchmark problems to test the convergence of the combination of the parareal algorithm with the embedded higher-order splitting schemes.

5.1. Test 1: Uni-directional coupling

We start with a scalar problem for the coupling of the heat equation with convection term with a fluid flow given by a convection equation:

$$\partial_t T = \nabla \cdot (K \nabla T) - \nabla \cdot (vT), \tag{5.1}$$

$$\partial_t v = -(v \cdot \nabla) v - \nabla p, \tag{5.2}$$

$$T(x, t_0) = T_0(x), \tag{5.3}$$

$$v(x, t_0) = v_0(x), \tag{5.4}$$

where the unknown temperature is T, v is the flow field of the temperature, and p is a given pressure. The spatial domain of this problem is the interval $[0, 1]$ with thermal conductivity $K = 0.1$, which we discretize by finite differences at $n + 1$ equidistant nodes $x_j = j/n$ ($j = 0, 1, \ldots, n$, $n = 100$). We assume we have homogeneous Neumann conditions at the interval endpoints. This results in the following system:

$$\partial_t \mathbf{T} = D_2 \mathbf{T} - D_1(\mathbf{T} \circ \mathbf{v}) + \mathbf{b}, \tag{5.5}$$

$$\partial_t \mathbf{v} = -\mathbf{v} \circ D_1 \mathbf{v} - D_1 \mathbf{p}, \tag{5.6}$$

where the matrices D_1, D_2 discretize the first and second order differential operators and $\mathbf{b}$ is a vector representing the boundary data. In a real-world problem, the pressure p would depend on the temperature vector T, but in this test we assume that the pressure is given by the function

$$p(t, x) = \mathbf{1}_{[0,1/3]}(t) \cdot 3\mathrm{e}3 \cdot t x^6 (1 - x)^6,$$

where $\mathbf{1}_{[0,1/3]}(t)$ is an indicator function for the time interval $[0, 1/3]$. This function creates a transport wave (a flow field) traveling in both directions from the midpoint of the interval towards its endpoints. The initial flow field is taken to be $v_0(x) = 0$ and for the initial temperature we have taken $T_0(x) = 16x^2(1 - x)^2$. The whole problem is integrated over the time domain $\Omega_T = [0, 1]$, which we have divided into 10 subdomains of equal length.

In order to evaluate the accuracy and performance of the coarse and fine integrators over one time-slice $[0, 0.1]$, we compare three different integrators for a general initial-value problem $u' = f(t, u)$, $u(0) = u_0$, namely,

- the implicit Euler method of order one (Euler1),
- a classical explicit Runge–Kutta method of order two (RK2 or midpoint method),
- the backward differentiation formula of order four (BDF4).

Note that the equation for the temperature is semi-linear. For semi-linear initial-value problems, $u' = Au + f(t, u)$, $u(0) = u_0$, we also test

- an exponential Runge–Kutta method of stiffness order two (expRK2 or exponential midpoint method),

which treats the stiff linear term Au exactly and hence the stability condition is only imposed by the mildly stiff nonlinear term. This method requires the evaluation of matrix exponentials $\exp(tA)$, or more precisely, the action of these exponentials on vectors $\exp(tA)v$. We have used a rational Chebyshev method for these evaluations.

We apply the above integrators to each of the two differential equations for T and v, where we have compared three different coupling techniques:

- A-B coupling, which is of order one and hence will only be tested in combination with Euler1,
- Strang coupling, which is of order two and hence will be tested in combination with the Runge–Kutta methods (RK2 and expRK2),
- iterative coupling in combination with the BDF4 integrator (note that in this example the coupling is uni-directional and hence there is only the need for exactly one splitting iteration).

In Fig. 5.1 we show the accuracy (the absolute error measured in the Euclidean norm) of the tested combinations as a function of the number of time-steps (top) and the number of function evaluations (below), respectively. As exact solution we have taken the result of a high accuracy integration.

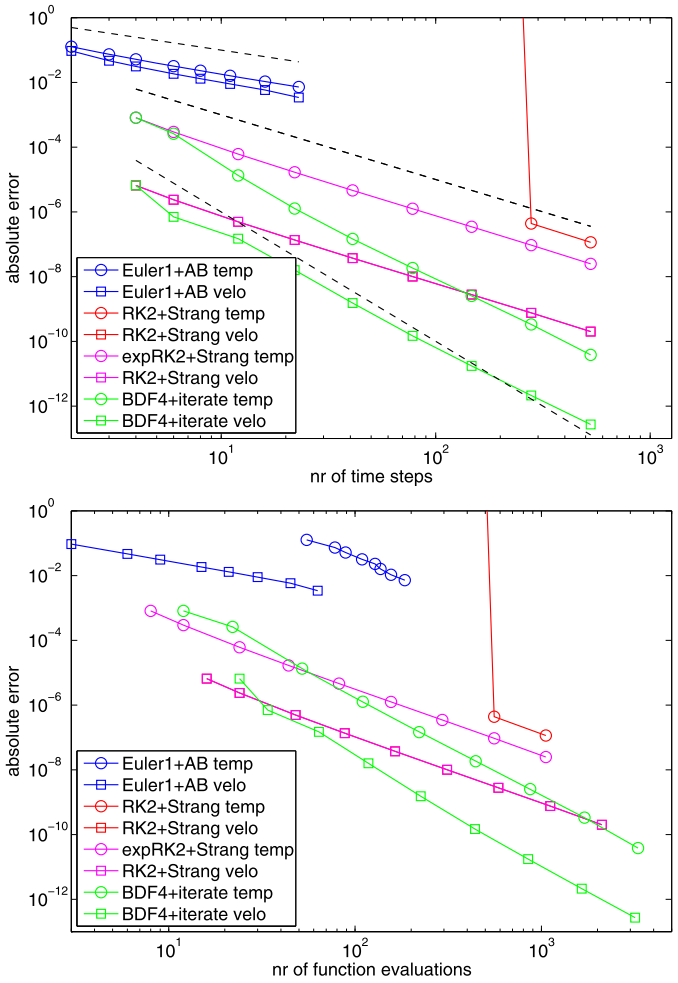

Fig. 5.1. Convergence of combinations of different time-stepping schemes with various coupling techniques for integration of the 1D model problem over a single time-slice $[0, 0.1]$. The coupling of the two equations is uni-directional. The upper plot shows the absolute error (Euclidean norm) as a function of the number of time-steps, with the dashed lines indicating convergence of orders one, two, and four. The lower plot shows the achieved absolute error (Euclidean norm) versus the required number of function evaluations.

In the number of function evaluations we have included for the implicit solvers (Euler1 and BDF4) the evaluations required by the Newton solver. We have used the m-file `nsoli.m`, a globally convergent Newton–Krylov solver given in [12], with an absolute and relative error tolerance tol proportional to tsteps^{-q}, written $\text{tol} \sim \text{tsteps}^{-q}$, where tsteps is the number of time-steps and q is the order of the integrator. For BDF4 ($q = 4$) we have also included the function evaluations for the higher-accuracy initialization of the four initial time-steps by expRK2 with a refined step-size h. Since expRK2 is of order two we have chosen $h \sim \sqrt{\text{tol}}$. Note that the number of function evaluations for the second equation (velocity) with the explicit methods and Strang splitting is twice as large as the number of function evaluations for the first equation (temperature), because in each time-step we make two half-steps with v and one full-step with T. Note also that the convergence of RK2 for the temperature equation is subject to stringent stability constraints and the convergence curve is reasonably low only for a large number of time-steps (or function evaluations). The expRK2 method does not have these stability constraints and shows order two convergence already for very small numbers of time-steps (functions evaluations).

The fact that the favorable stability properties of expRK2 allow for larger step-sizes makes this method (in conjunction with Strang coupling) predestined to be the coarse integrator for parareal. In order to achieve a high accuracy with few function evaluations we will use BDF4 with iterative coupling as the fine integrator.

The convergence of parareal is depicted in Fig. 5.2. Each convergence curve corresponds to the error of the computed temperature T (left plot) and velocity v (right plot) measured at each of the 11 coarse time points $T_0, T_1, \ldots, T_N$, where the abscissae indicate the parareal iteration index k. The error is measured in the Euclidean norm compared to a high accuracy reference solution. Note that necessarily after N iterations, parareal reaches the accuracy of the fine propagator, the result of which was also used as the reference solution for computing the error. However, the stagnation level of the accuracy is reached already after $k = 4$ parareal iterations, which means that we achieved a parallel speedup of 10/4 (when neglecting the cost of the coarse integrator). Note that—to avoid adding more complications to the interpretation of our numerical

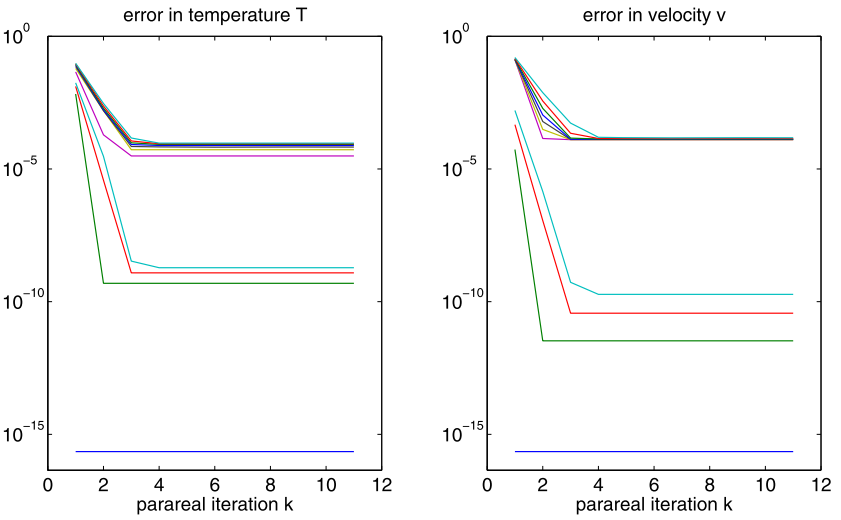

Fig. 5.2. Convergence of parareal applied to the 1D model problem. The coupling is uni-directional. The left plot shows the error in the temperature variable (Euclidean norm, compared to a high accuracy reference solution) at each of the 11 points of the time grid (the lowest constant curve corresponds to time $t = 0$, where the initial value is given) after $k = 1, 2, \ldots$ parareal iterations. On the right we show the error in the velocity variable.

results—for all examples we have used a fixed step-size in the integrators, which results in the different stagnation levels of the accuracies. With an adaptive step-size fine integrator, one would certainly try to achieve an absolute error of about, say, 10^{-3}, avoiding the reduction below 10^{-9} in time-slices where the solution is very smooth.

5.2. Test 2: Bi-directional coupling

We reconsider the previous test problem but now with a pressure function depending on the temperature T,

$$p(T) = 0.2 \cdot T + 0.2.$$

Therefore we now have a bi-directional coupling between (5.1) and (5.2). For the iterative splitting method this results in the system

$$\partial_t \mathbf{T}_i = D_2 \mathbf{T}_i - D_1(\mathbf{T}_i \circ \mathbf{v}_{i-1}) + \mathbf{b}, \quad \mathbf{T}_i t^n = \mathbf{T}(t^n),\ t \in [t^n, t^{n+1}], \tag{5.7}$$

$$\partial_t \mathbf{v}_i = -\mathbf{v}_{i-1} \circ D_1 \mathbf{v}_i - D_1 \mathbf{p}(T_i), \quad \mathbf{v}_i t^n = \mathbf{v}(t^n),\ t \in [t^n, t^{n+1}], \tag{5.8}$$

where $i = 1, 2, \ldots, I$ and $I \in \mathbb{N}^+$ is a fixed number. Further we assume the starting value $v_0(t) = v(t^n)$ or an initial guess $v_0(t) = v_{\text{initial}}(t)$. The approximate solutions are $\mathbf{T}(t^{n+1}) = \mathbf{T}_I t^{n+1}$ and $\mathbf{v}(t^{n+1}) = \mathbf{v}_I t^{n+1}$ after I iterative steps.

As in the previous test, we have used the Newton solver `nsoli` for the implicit time-stepping methods and we have chosen the stopping tolerance tol depending the number of time-steps tsteps as $\text{tol} \sim \text{tsteps}^{-q}$, as before. The achieved accuracy (absolute error in the Euclidean norm) as a function of the number of time-steps is shown in Fig. 5.3 (top).

We observed that for every time-step, only 2–3 iterative splitting-scheme iterations are required to satisfy the imposed error criterion. However, each call to the Newton solver `nsoli` requires 3–7 function calls for solving the implicit equation for the new temperature or velocity iterate to prescribed error tolerance. The average number of function calls per time-step for the temperature equation is therefore above four, which results in a worse work-precision curve for the iterative splitting method when the number of function evaluations is taken as the measure (see Fig. 5.3, bottom). In this example, only for very high accuracies (below 10^{-7}, which is far below our spatial discretization error) can we expect that the iterative splitting approach outperforms the expRK2 scheme with Strang coupling.

The convergence of parareal for the temperature and velocity components is illustrated in Fig. 5.4. As before, we have partitioned the time interval $[0, 1]$ into ten time-slices of equal length. As coarse integrator, we have used one time-step per time-slice of the expRK2 method with Strang coupling. The fine integrator is 200 time-steps of BDF4 with iterative coupling. Note that after about eight iterations of parareal, we reach the final stagnation level of the error (Euclidean norm), which means that we can expect a parallel speedup of 10/8, provided that the cost for the coarse integrator is negligible.

Remark 5.1. We have obtained an acceleration of our solver process with a speedup of 10/8. The same speedup can be obtained in experiments with two- or three-dimensional problems. Here the speedup can be improved by using an iterative scheme to obtain a decomposition of the multidimensional problem, see [6].

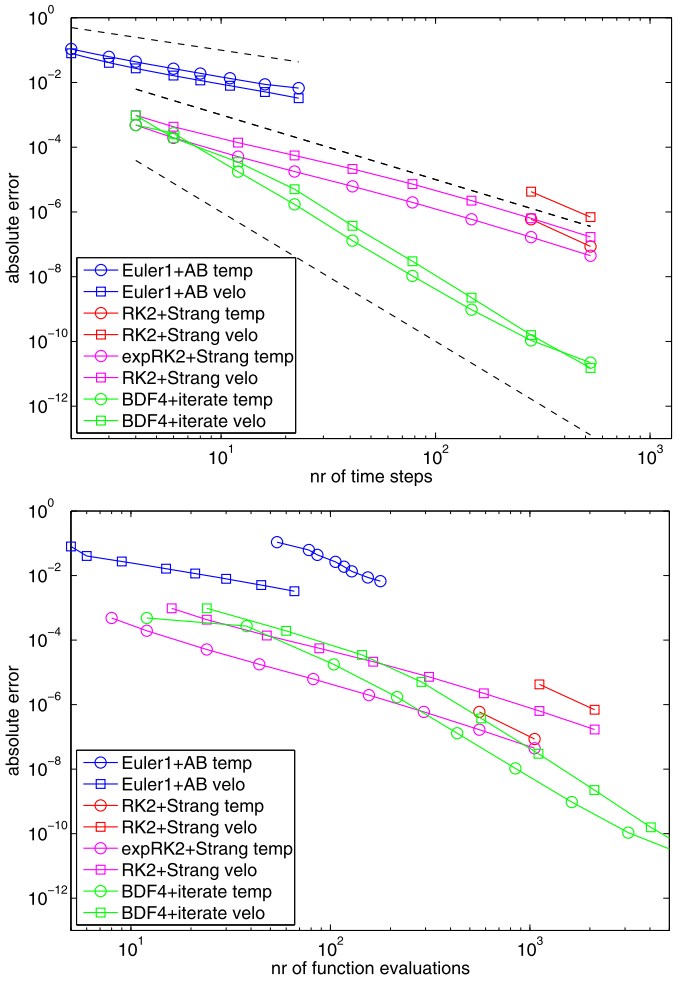

Fig. 5.3. Convergence of combinations of different time-stepping schemes with various coupling techniques for integration of the 1D model problem over a single time-slice $[0, 0.1]$. The coupling of the two equations is bi-directional. The dashed lines on the left indicate convergence of orders one, two, and four.

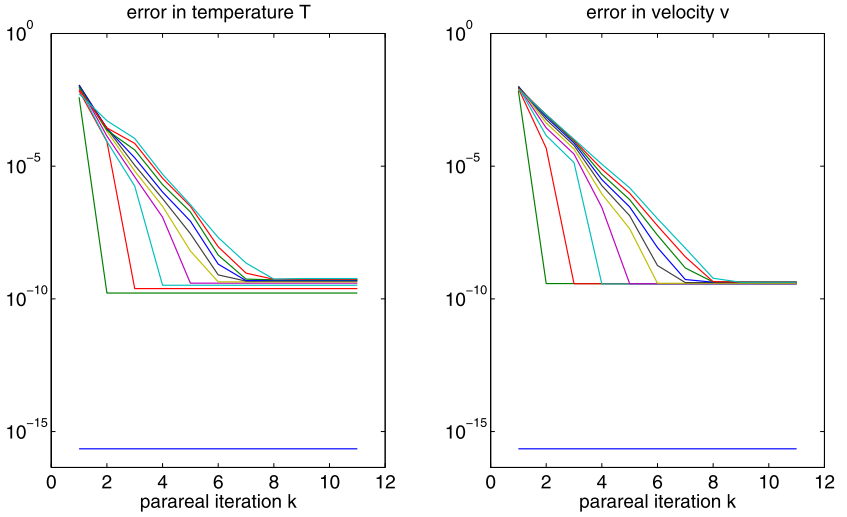

Fig. 5.4. Convergence of parareal applied to the 1D model problem with bi-directional coupling. The left plot shows the error of the temperature variable at each of the 11 points of the time grid (the lowest constant curve corresponds to time $t = 0$, where the initial value is given) after $k = 1, 2, \ldots$ parareal iterations. On the right we show the error of the velocity variable.

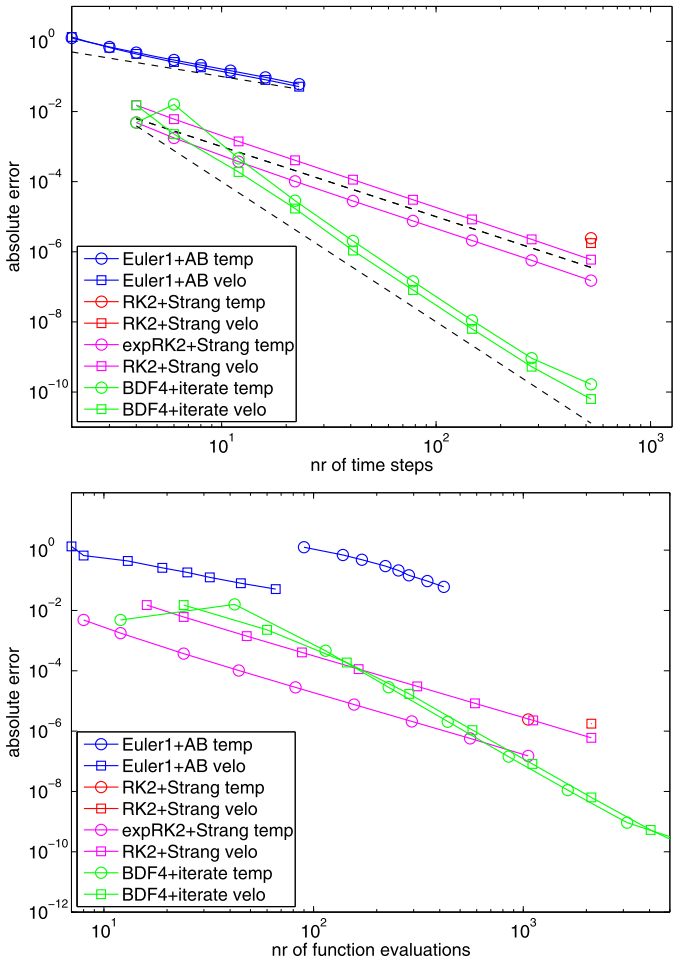

Fig. 5.5. Convergence of combinations of different time-stepping schemes with various coupling techniques for integration of the 2D model problem (with bi-directional coupling) over a single time-slice $[0, 0.1]$. The coupling of the two equations is bi-directional. The dashed lines on the left indicate convergence of orders one, two, and four.

5.3. Test 3: Bi-directional coupling in two dimensions

We extend the previous example to a two-dimensional problem coupling a heat equation with convection term and a fluid flow given by a convection equation:

$$\partial_t T = \nabla \cdot (K \nabla T) - \nabla \cdot \mathbf{v} T, \tag{5.9}$$

$$\partial_t \mathbf{v} = -(\mathbf{v} \cdot \nabla) v - \nabla p(T), \tag{5.10}$$

$$T(x, y, t_0) = T_0(x, y), \tag{5.11}$$

$$\mathbf{v}(x, y, t_0) = \mathbf{v}_0(x, y), \tag{5.12}$$

where the unknown temperature is T, $\mathbf{v}$ is the flow field of the temperature, and the pressure is

$$p(t, x, y, T) = \mathbf{1}_{[0,1/3]}(t) \cdot 1\text{e}4 \cdot t x^2 (1-x)^6 y^2 (1-y)^6 + 0.1 \cdot T.$$

The spatial domain of this problem is the square $[0, 1] \times [0, 1]$ with thermal conductivity $K = 0.1$, discretized by finite differences on a regular grid $x_j = j/n$, $y_k = k/n$ $(j, k = 0, 1, \ldots, n,\ n = 100)$. We assume we have homogeneous Neumann conditions on the boundary. The initial flow field is $\mathbf{v}_0(x, y) = \mathbf{0}$ and for the initial temperature we have taken $T_0(x, y) = 16x^2(1-x)^2 y^2 (1-y)^2$.

As in the previous examples, we first investigate the accuracy of various coarse and fine integrators over a single time-slice $[0, 0.1]$. The achieved accuracy (absolute error in the Euclidean norm) as a function of the number of time-steps is shown in Fig. 5.5 (top).

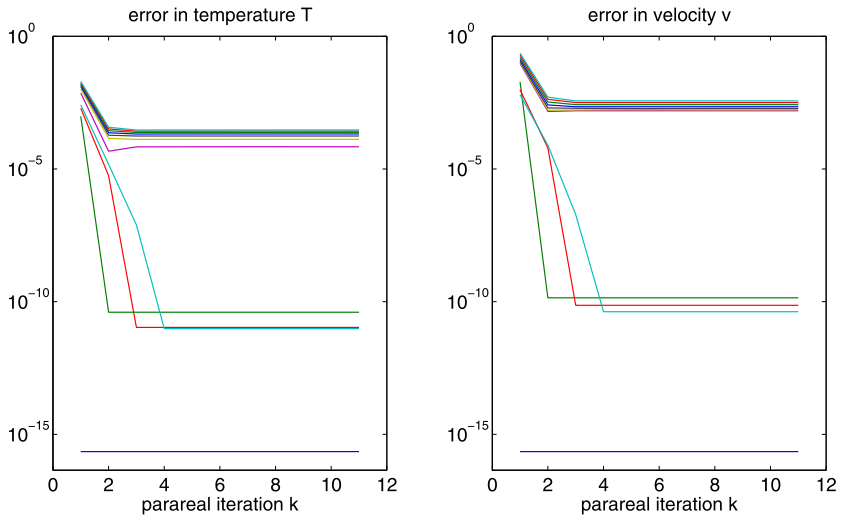

Fig. 5.6. Convergence of parareal applied to the 2D model problem with bi-directional coupling. The left plot shows the error of the temperature variable at each of the 11 points of the time grid (the lowest constant curve corresponds to time $t = 0$, where the initial value is given) after $k = 1, 2, \ldots$ parareal iterations. On the right we show the error of the velocity variable.

As before, we observe that for every time-step only 2–3 iterative splitting-scheme iterations are required to satisfy the imposed error criterion. However, each call to the Newton solver `nsoli` requires 3–7 function calls for solving the implicit equation for the new temperature or velocity iterate to prescribed error tolerance. The average number of function calls per time-step for the temperature equation is therefore above four, which results in a worse work-precision curve for the iterative splitting method when the number of function evaluations is taken as the measure (see Fig. 5.5, bottom). In this example, only for accuracies below about below 10^{-6} (which is far below our spatial discretization error) can we expect that the iterative splitting approach outperforms the expRK2 scheme with Strang coupling.

The convergence of parareal for the temperature and velocity components is illustrated in Fig. 5.6. As coarse integrator we have used one time-step per time-slice of the expRK2 method with Strang coupling. The fine integrator is 200 time-steps of BDF4 with iterative coupling. The convergence behavior of parareal is very similar to that observed in the first 1D example. This indicates that the convergence of both the coarse and fine integrator does not depend significantly on the dimension of the problem. One explanation for this observation is the fact that the coarse integrator is robust to the dominating linear diffusion part (because this part is treated exactly by the exponential integrator, and the length of the spectral interval depends only linearly on the number of space dimensions), and the fine integrator is an implicit method (therefore stable without any restrictions on the step-size). After about four iterations of parareal we reach the final stagnation level of the error (Euclidean norm), which means that we can expect a parallel speedup of 10/4, provided that the cost for the coarse integrator is negligible.

6. Conclusion

We have presented an iterative coupling method for a heat transport and flow problem arising in a technical application. We have discussed existing coupling methods and compared them to higher-order splitting schemes. Additional parallel speedup in the time-domain can be obtained by the parareal iteration, although we have observed that the speedup depends heavily on the degree of coupling between both equations. The numerical examples were carried out with a fixed step-size in the coarse and fine integrators and the effect of adaptive step-size control should be subject of further investigation. A derivation of a closed framework of parallel splitting schemes will also be the subject of future work.

References

[1] I. Farago, J. Geiser, Iterative operator-splitting methods for linear problems, Int. J. Comput. Sci. Eng. 3 (4) (2007) 255–263.
[2] M.J. Gander, Analysis of the parareal algorithm applied to hyperbolic problems using characteristics, Bol. Soc. Esp. Mat. Apl. SeMA 42 (2008) 21–35.
[3] M.J. Gander, E. Hairer, Nonlinear convergence analysis for the parareal algorithm, in: Proceedings of the 17th International Domain Decomposition Conference, Springer-Verlag, Berlin, 2008, pp. 45–56.
[4] M.J. Gander, S. Vandewalle, Analysis of the parareal time-parallel time-integration method, SIAM J. Sci. Comput. 29 (2) (2007) 556–578.
[5] J. Geiser, Iterative operator-splitting methods with higher order time-integration methods and applications for parabolic partial differential equations, J. Comput. Appl. Math. 217 (2008) 227–242.
[6] J. Geiser, Chr. Kravvaritis, A domain decomposition method based on iterative operator splitting method, Appl. Numer. Math. 59 (2009) 608–623.
[7] J. Geiser, Iterative operator-splitting methods for nonlinear differential equations and applications, Numer. Methods Partial Differential Equations 27 (5) (2011) 1026–1054.
[8] J. Geiser, Iterative splitting methods for differential equations, in: Magoules, Lai (Eds.), Numerical Analysis and Scientific Computing Series, Chapman & Hall/CRC, 2011.

[9] E. Hairer, G. Wanner, Solving Ordinary Differential Equations II, Springer Ser. Comput. Math., vol. 14, Springer-Verlag, Berlin, 1996.
[10] J. Kanney, C. Miller, C.T. Kelley, Convergence of iterative split-operator approaches for approximating nonlinear reactive transport problems, Adv. Water Resources 26 (2003) 247–261.
[11] K.H. Karlsen, N. Risebro, An operator splitting method for nonlinear convection-diffusion equation, Numer. Math. 77 (3) (1997) 365–382.
[12] C.T. Kelly, Iterative Methods for Linear and Nonlinear Equations, Frontiers Appl. Math., SIAM, Philadelphia, PA, USA, 1995.
[13] P. Lindelöf, Sur l'application des methodes d'approximations successives a l'etude de certaines equations differentielles ordinaires, J. Math. Pures Appl. 4 (9) (1893) 217–271.
[14] P. Lindelöf, Sur l'application des methodes d'approximations successives a l'etude des integrales reelles des equations differentielles ordinaires, J. Math. Pures Appl. 4 (10) (1894) 117–128.
[15] J.L. Lions, Y. Maday, G. Turincini, A "parareal" in time discretization of PDEs, C. R. Math. Acad. Sci. Paris 332 (2001) 661–668.
[16] L. Prandtl, O.G. Tietjens, Fundamentals of Hydro- and Aeromechanics, Dover, New York, 1957.
[17] B. Sportisse, An analysis of operator splitting techniques in the stiff case, J. Comput. Phys. 161 (2000) 140–168.
[18] G. Strang, On the construction and comparison of difference schemes, SIAM J. Numer. Anal. 5 (1968) 506–517.
[19] Ricardo Software, VECTIS, three-dimensional fluid dynamics program, http://www.ricardo.com/What-we-do/Software/Products/VECTIS/.
[20] S. Vandewalle, Parallel Multigrid Waveform Relaxation for Parabolic Problems, B.G. Teubner, Stuttgart, 1993.
[21] J.R. Welty, C.E. Wicks, R.E. Wilson, Fundamentals of Momentum, Heat, and Mass Transfer, 2nd edition, Wiley, New York, 1976.
[22] E. Zeidler, Nonlinear Functional Analysis and its Applications. II/B Nonlinear Monotone Operators, Springer-Verlag, Berlin, 1990.